
12th ANNUAL WORKSHOP 2016

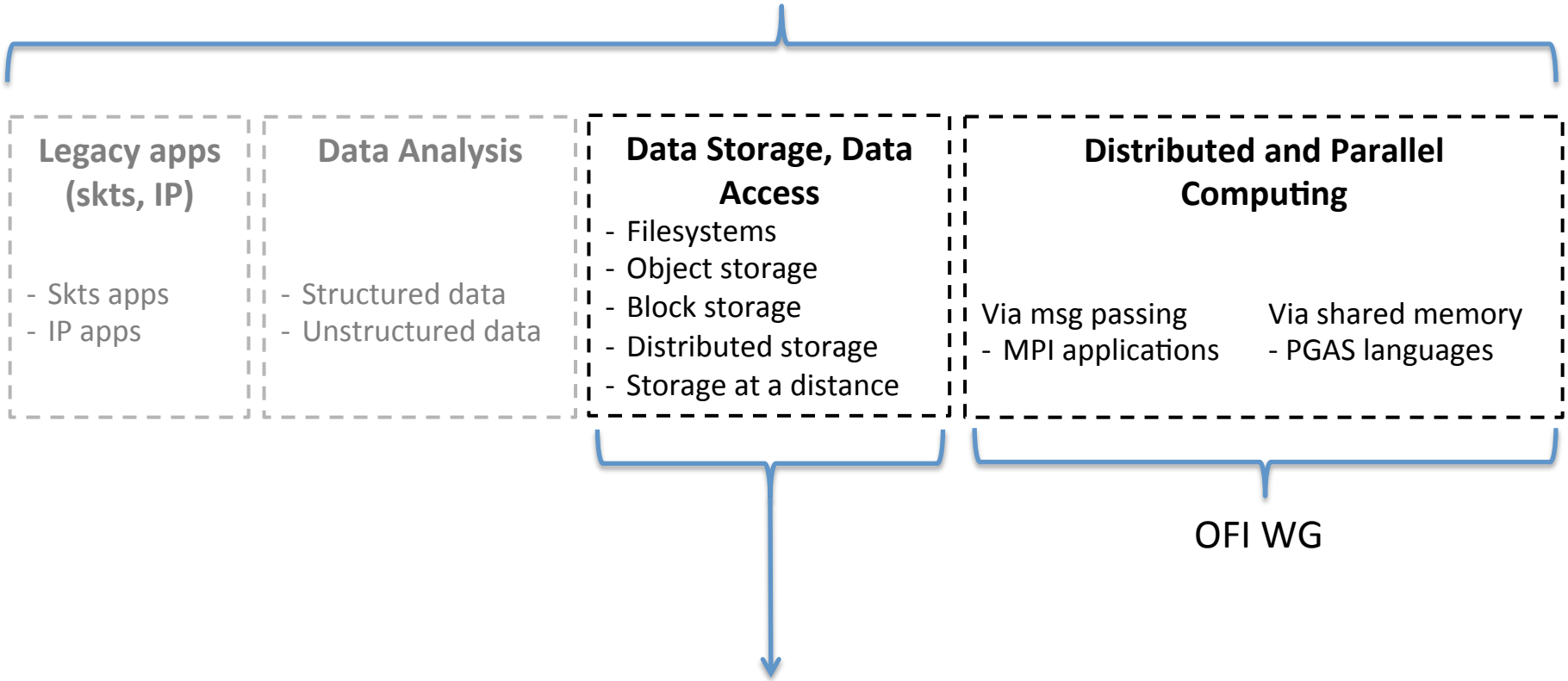
# APIs for an NVM World

Paul Grun

Cray, Inc

April 6, 2016

# OFI Project




**The work described in this session began as part of the DS/DA Working Group's investigation**

**How does the emergence of NVM impact the network stack?**

**Specifically, the API?**

**What do fabric consumers need?**

**To answer that, we need to think a little bit about how NVM is used**

# SCOPE

- **Use cases**
  - NVM as a target of memory operations
  - NVM as a target of I/O operations
- **Locality**
  - A device attached to an I/O bus (PCIe) or a memory channel
  - A remote device accessed over a network
- **Modes**
  - User mode
  - Kernel mode

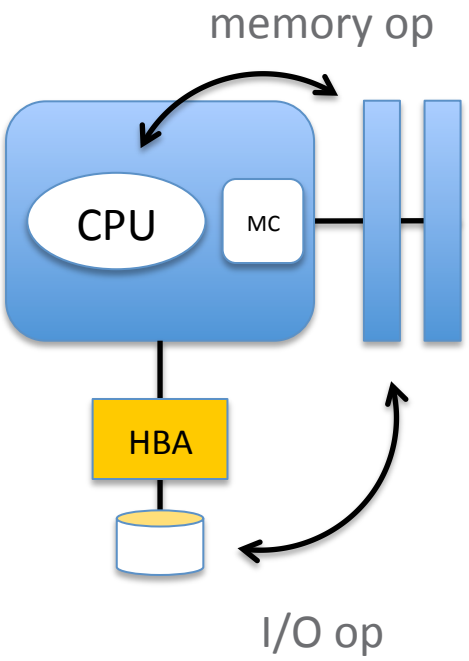
Not all of the above will turn out to be in scope for the DS/DA Work Group

# I/O? MEMORY?

- An extent (block) of data identified by a protocol-specific identifier (LBA) is transferred between memory and a storage device

**Memory operation e.g. Load/Store**

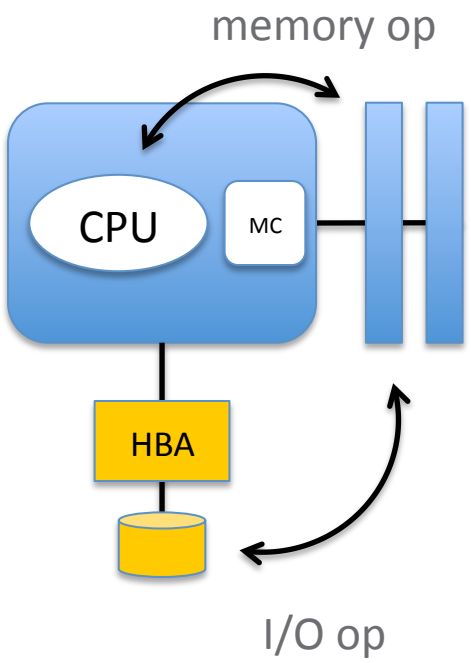
- Data is stored from a CPU register to a memory location



# TRANSACTION MODEL

**I/O**
- Client/server request/response protocol
- Completion occurs when the server sends a completion message

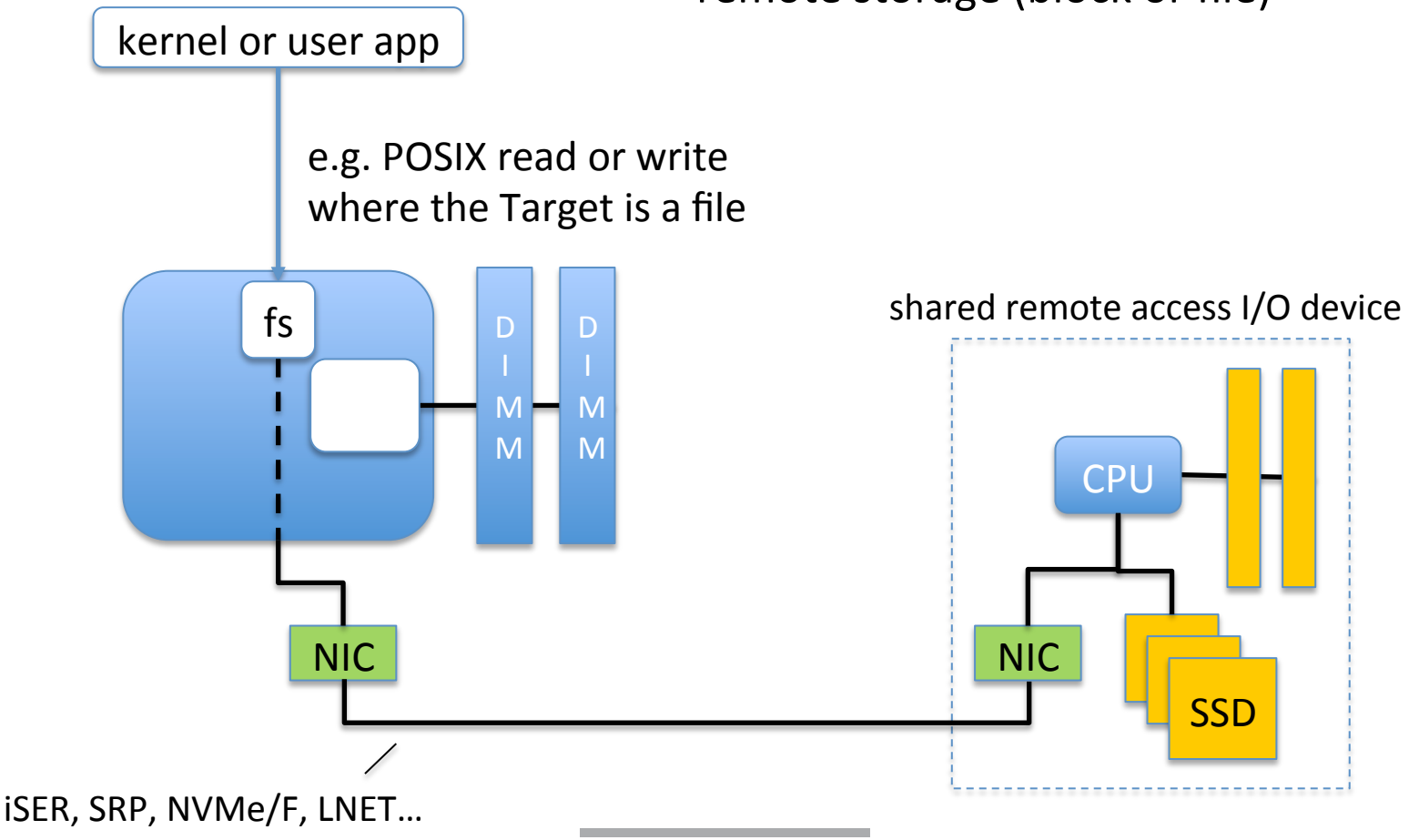
**Memory operations**
- A user reads or writes data to/from memory
- Completion occurs when the write is acknowledged




A significant difference from the application perspective

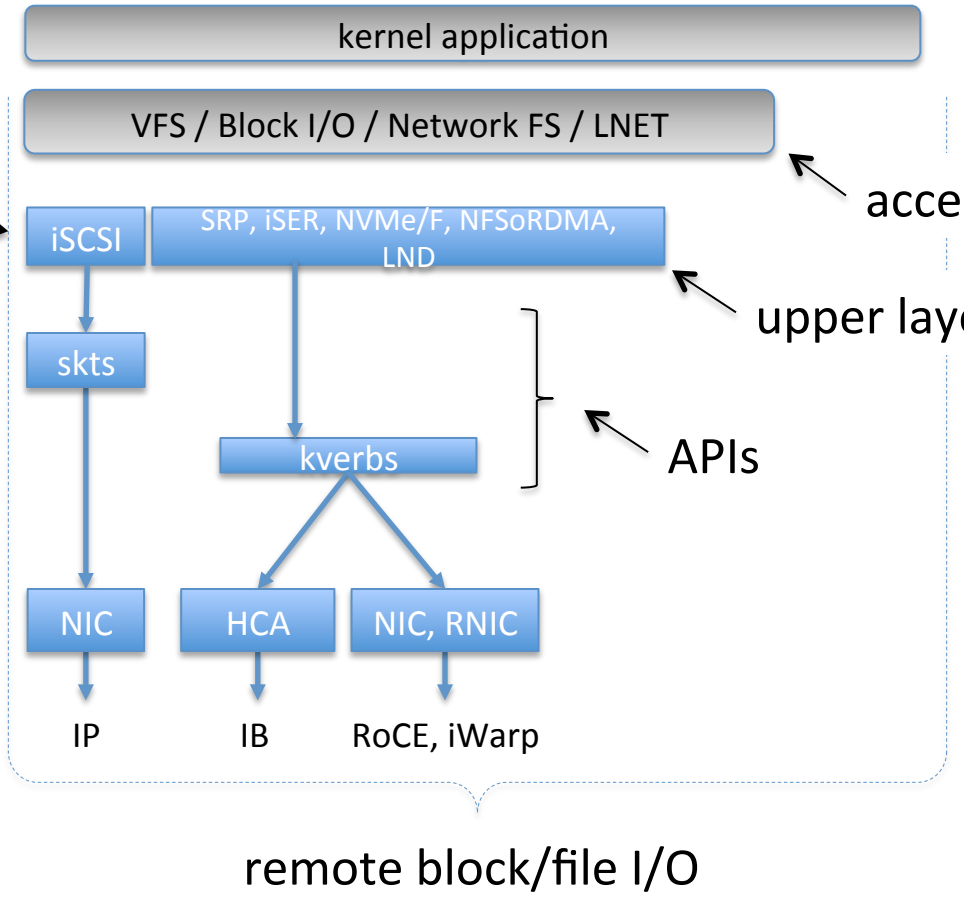
# USE CASE: STORAGE

Well-understood methods for accessing remote storage (block or file)



# ACCESS METHODS FOR STORAGE

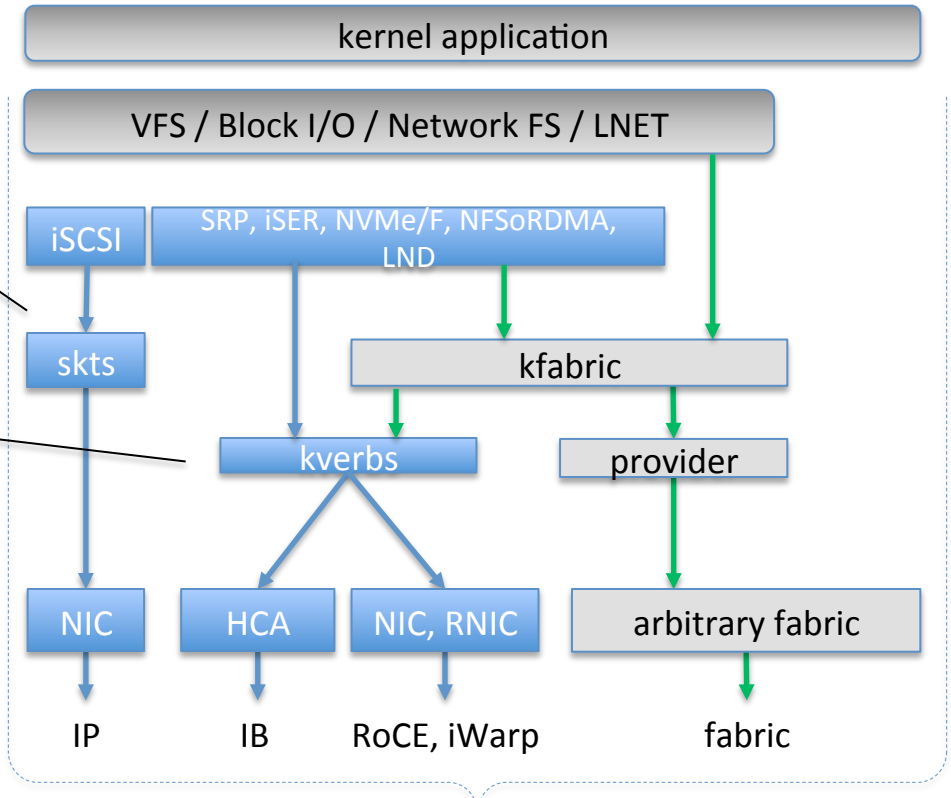
Synchronization occurs at this layer when the server returns ending status



# ACCESS METHODS FOR STORAGE

reliable sockets semantics don't map well to reliable messaging operations

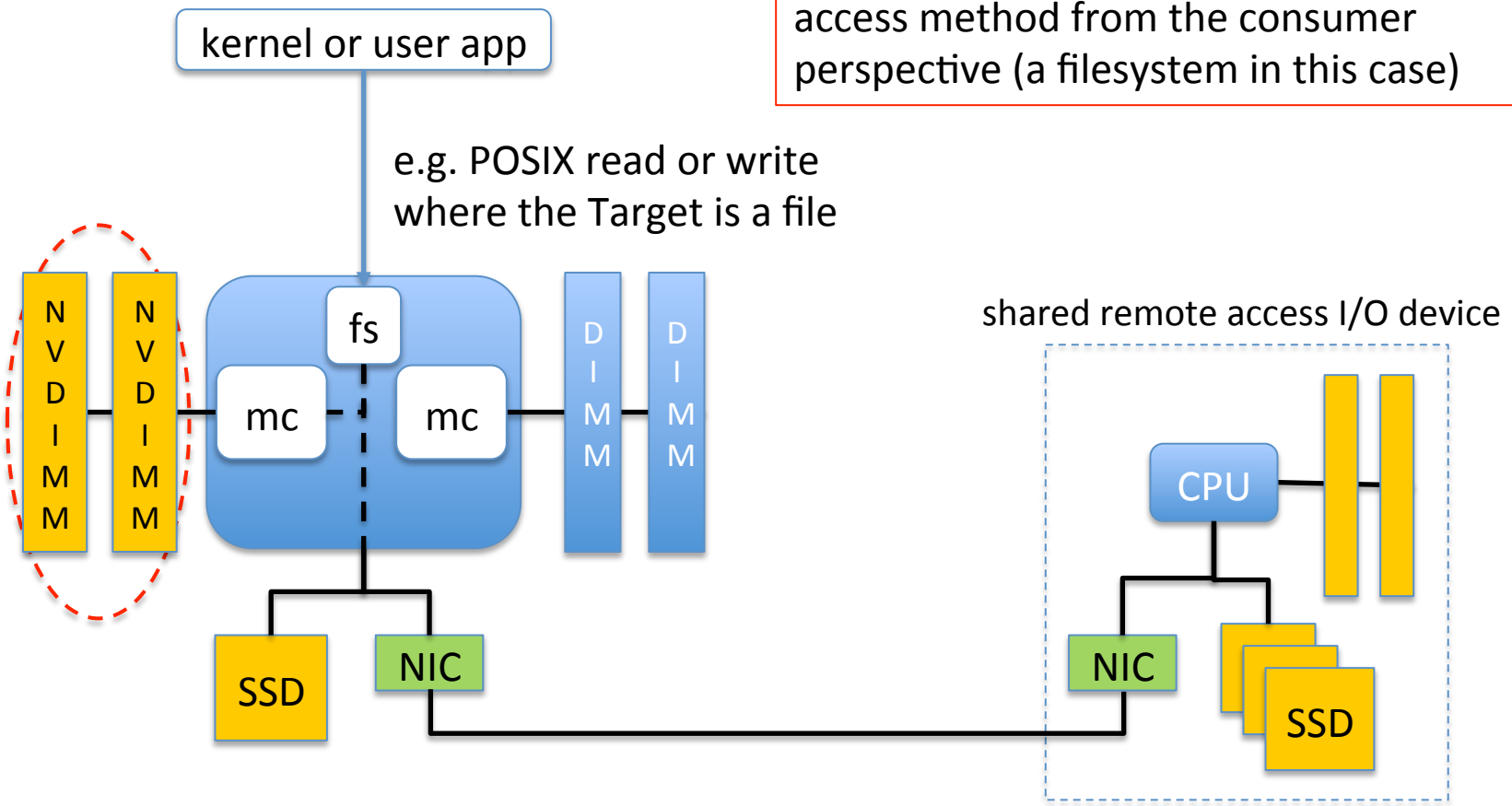
assumes a 'QP based' fabric



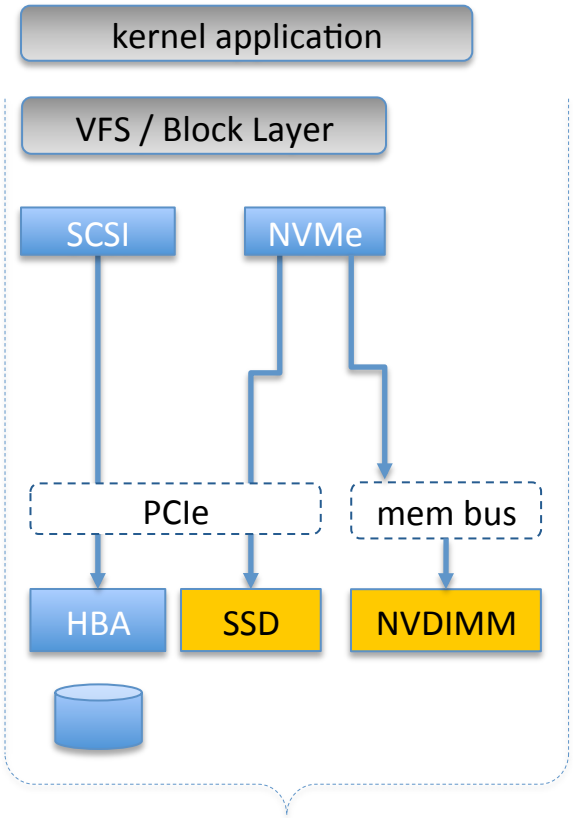

What happens when new fabrics emerge?

# USE CASE: STORAGE

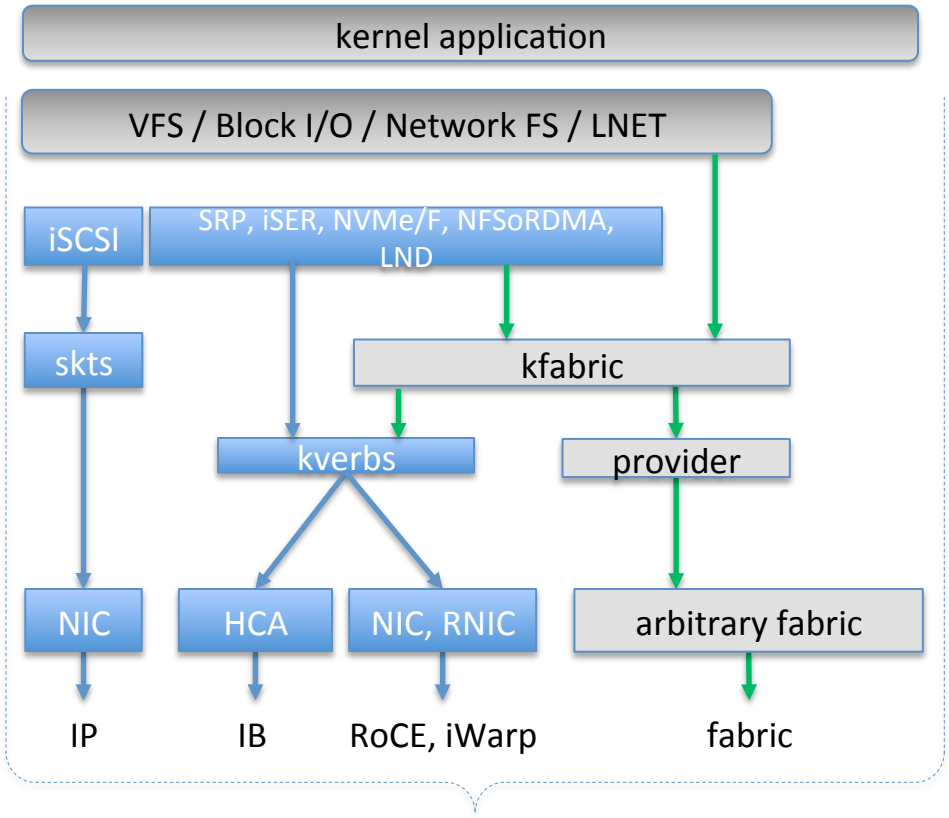
Slightly more interesting in the case of local NVDIMM, but no substantial difference in access method from the consumer perspective (a filesystem in this case)



# ACCESS METHODS FOR STORAGE

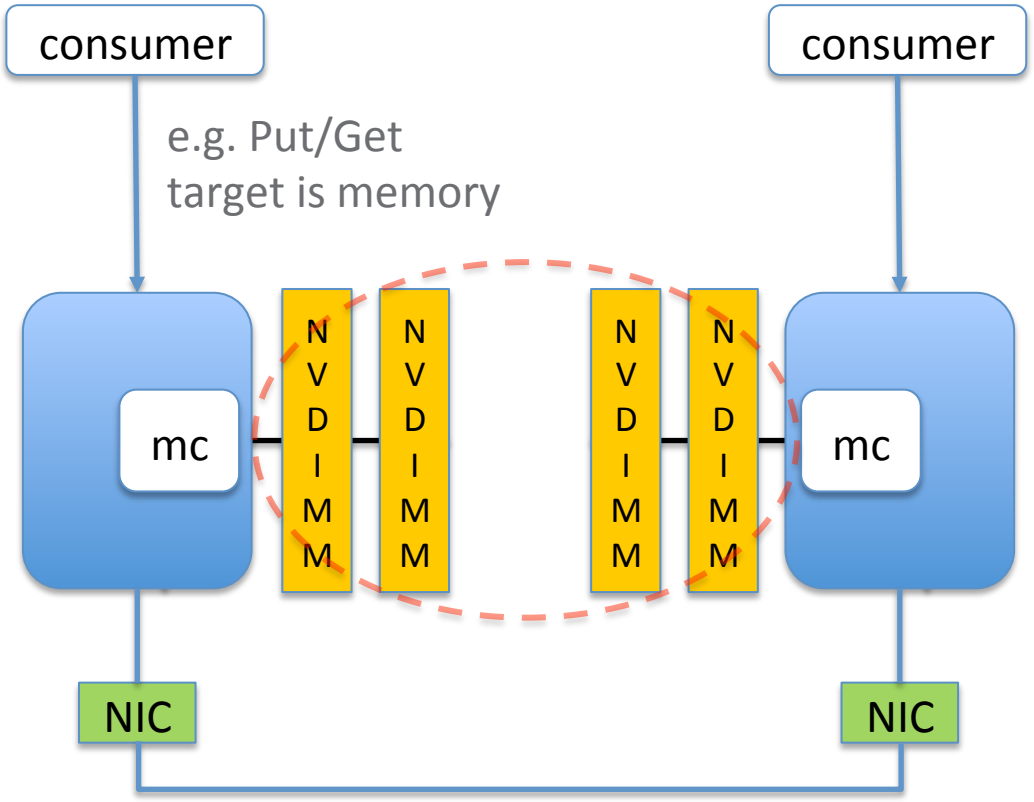



local block I/O




remote block/file I/O

# USE CASE: PERSISTENT MEMORY



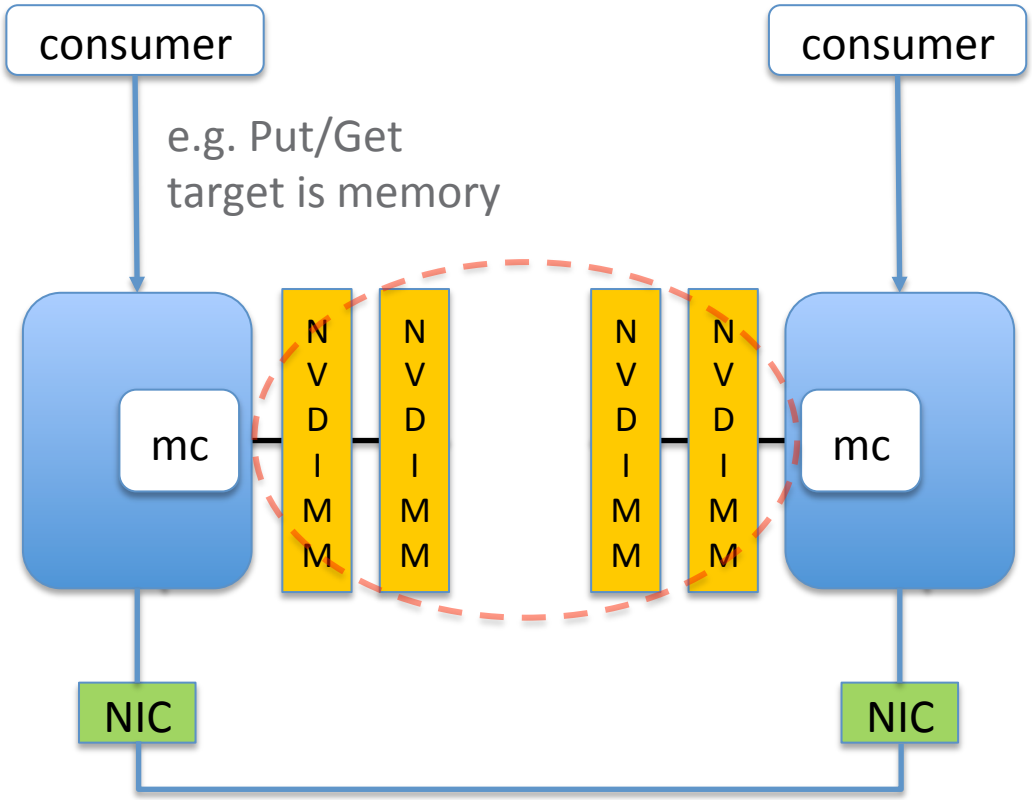

Something a bit different

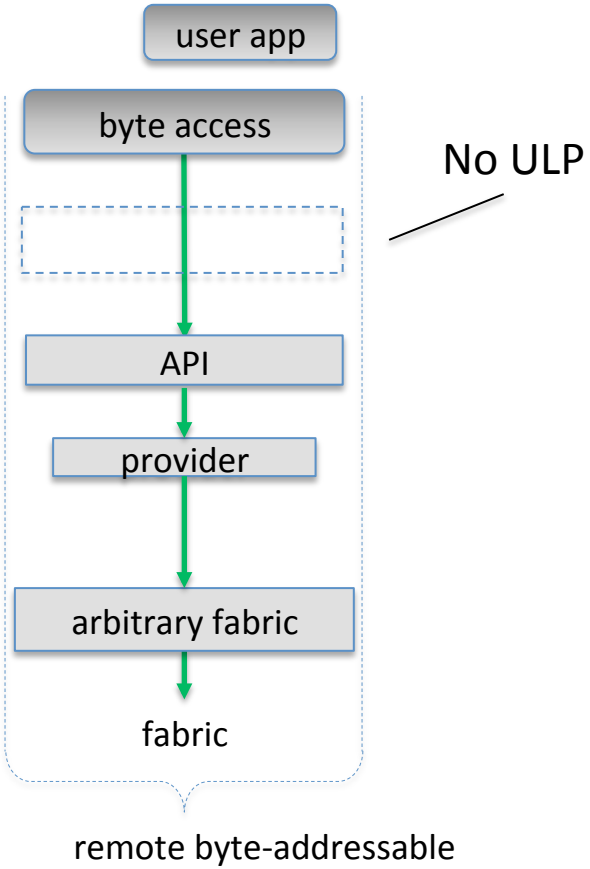
- Consumer treats access to remote PM as a memory read or write
- emphasis on single ended operations
- May be synchronous or asynchronous
- Completions must comprehend the notion of a persistence domain

example: PGAS, logfile updates…

"PMoF" – byte level interface

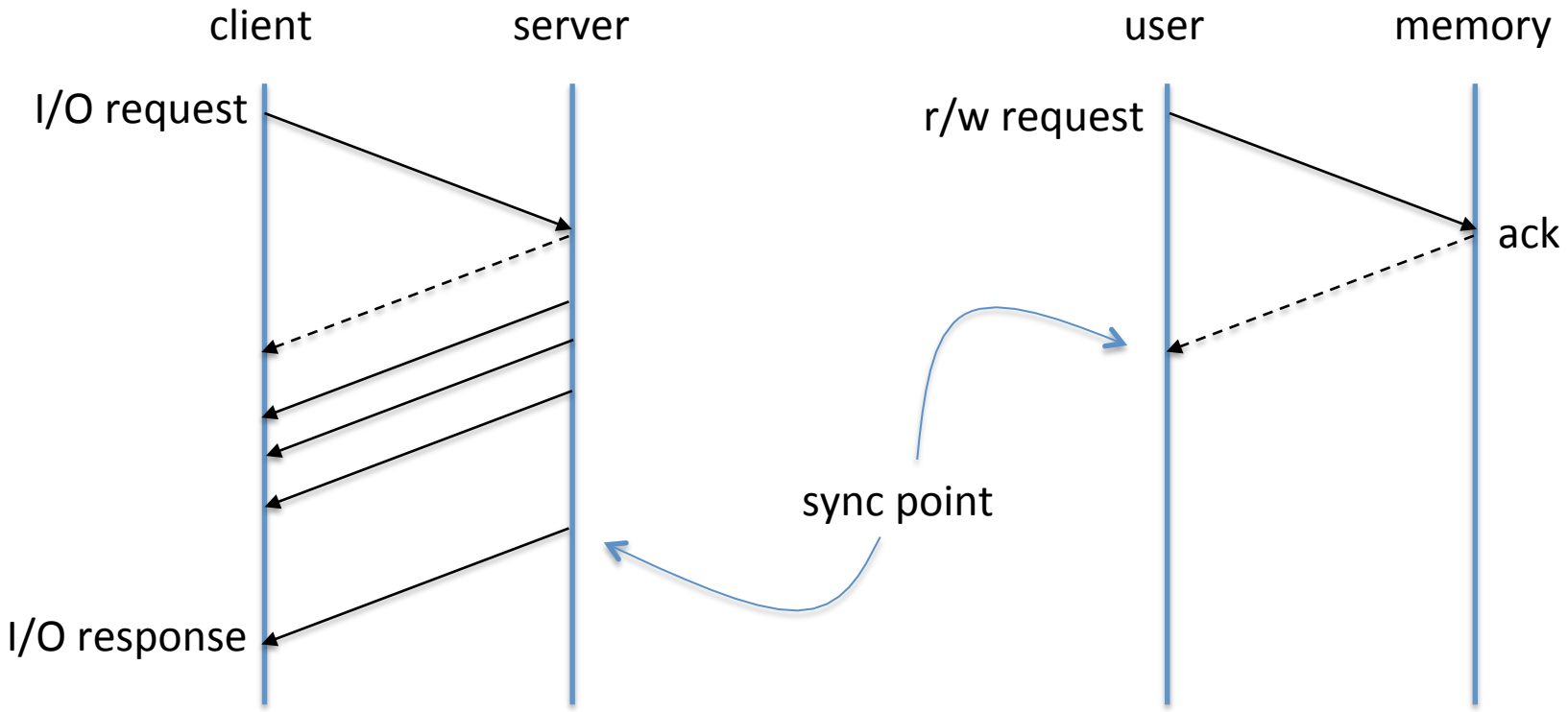
# USER MODE ACCESS TO PERSISTENT MEMORY




example: PGAS, logfile updates…

# COMPLETION/SYNCHRONIZATION MODELS



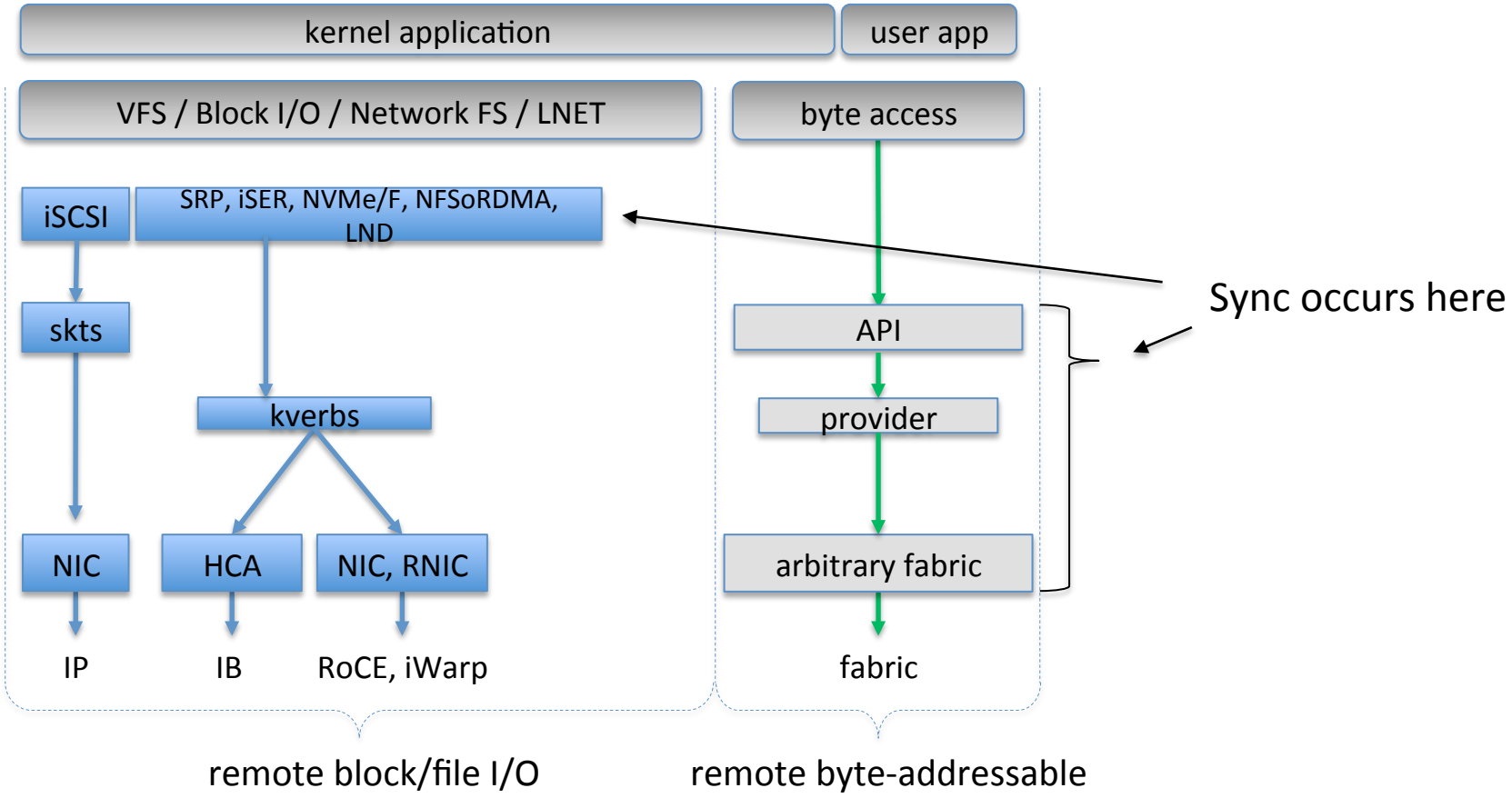

I/O transaction

Synchronization is accomplished by the I/O protocol

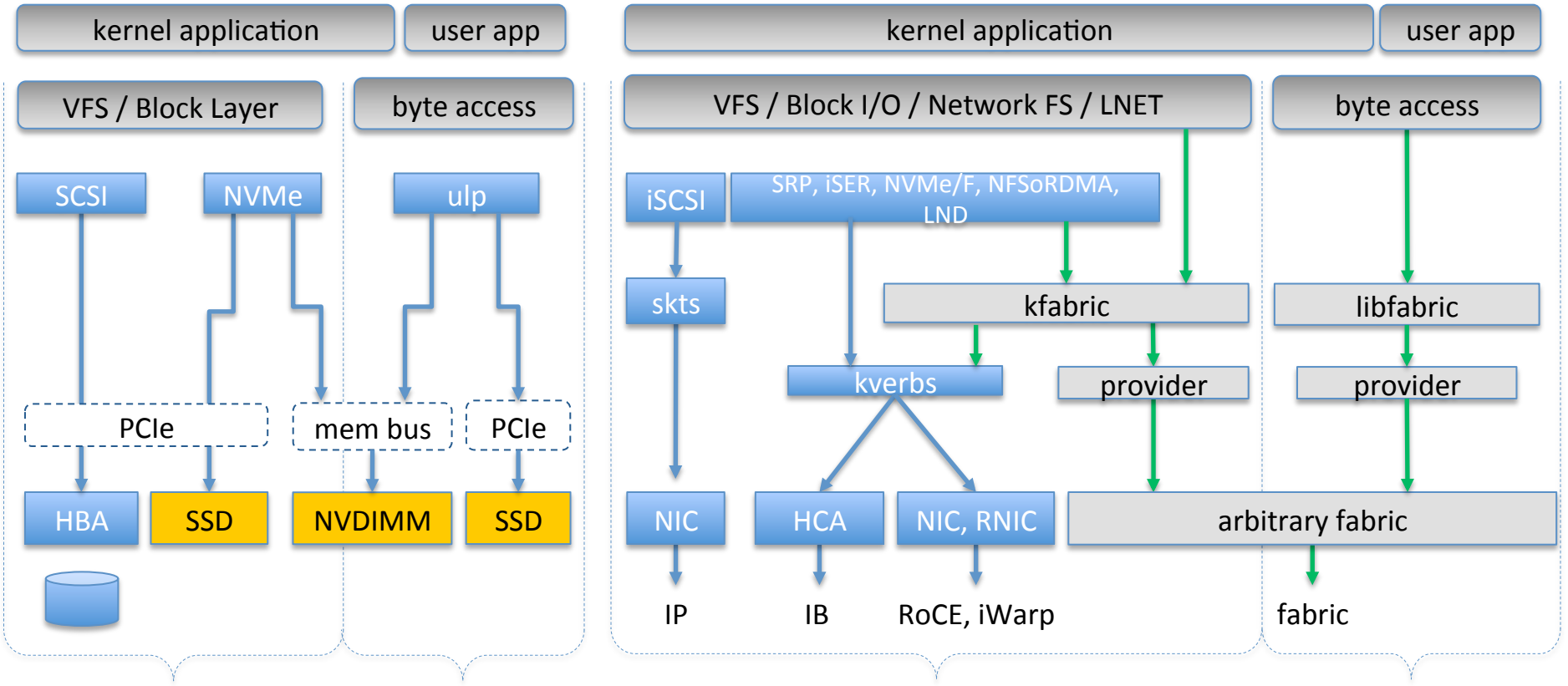
Memory operation

Synchronization occurs in hardware

# I/O VS REMOTE MEMORY



# POSSIBLE OFI-BASED I/O STACK



# TAKEAWAYS

- **Focus on both remote I/O and remote memory use cases**

**On the I/O side:**

- **As new fabrics emerge, existing ULPs will have to adjust**
- **Anticipate consumer needs to support them**

**On the memory side:**

- **Remote persistent memory has unique transactional requirements. These will impact the design of the API**

OPENFABRICS
ALLIANCE

12th ANNUAL WORKSHOP 2016

# THANK YOU